Interface System and Method of Building Rules and Constraints for a Resource Scheduling System

INVENTORS

Tiffany D. Boehmer, Gal Josefsberg, and Gary Todd Aoki

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is related to copending U.S. Application Serial No. 09/056,219 by Andre et al., filed April 7, 1998, and commonly assigned, the specification of which is incorporated herein by reference in its entirety.

FIELD OF THE INVENTION

This invention relates to computer systems and computer-implemented methods for generating and optimizing schedules for a collection of personnel. In particular, the invention relates to methods for optimizing personnel schedules to satisfy variable staffing requirements and shift constraints.

BACKGROUND OF THE INVENTION

The task of creating an efficient schedule for a collection of personnel is known to be a very complex constraint satisfaction problem. The problem becomes even more complex when the number of personnel is large, or when there are variable constraints upon the shifts such as the number of breaks within an person's shift, the scheduling of the breaks within a shift, and the lengths of the breaks.

Traditionally, scheduling is performed manually. Because the manual preparation of schedules is very laborious, however, computerized scheduling methods have been devised to perform the task of scheduling. Although computerized scheduling is far faster than manual scheduling, it often does not produce schedules that make the most efficient use of personnel. One major reason for the difficulty in producing efficient schedules is that the scheduling problem is a computational problem of type NP-complete, i.e. the only known algorithms to solve the problem require an amount of time that grows exponentially with the size of the problem. There are no known algorithms, therefore, which can perfectly solve the scheduling problem in less than exponential time. The problem of generating a good schedule, therefore, requires computational techniques for maximizing the schedule's efficiency while minimizing the amount of computational time, and inherently involves trade-offs that are peculiar to each specific application.

Traditional resource scheduling software packages include little or no mechanisms for the user to specify customized rules of a highly expressive nature. Instead, the user must simply follow a strict workflow imposed by the software or the user must specify constraints in great detail, requiring for example specifications of every particular personnel and every particular case of the rule individually.

Microsoft Outlook™ and other software products include simple interfaces for creating meaningful sentences using drag-and-drop actions. The structure of the "sentences" is not dynamically based on inputs by the user in general. They are fixed branches

of possible values and sets of values. A real-time database lookup feature is not part of these systems.
U.S. Patent 5,909,678 teaches building statements by dragging and dropping templates and components, and is specific to a system with selectable objects (for the "blanks") only.

U.S. Patent 5,682,542 is very specific to construction of statements containing noun and verb objects only.

U.S. Patents 5,590,322 and 5,592,668 teach designing queries for an information system, including a drag-and-drop method for creating the queries, but include only a system that maps the query to a specific internal structure (the fact tree representation).

U.S. Patent 5,584,024 teaches a narrow database query system. However, the system includes a user interface allowing a user to select from tables to create queries that are guaranteed to be syntactically and semantically correct, and thus conform to a strict grammar.

U.S. Patent 4,829,423 generally teaches any interface for users to create sentences from a specified grammar, including a means for generating all possible choices midway through the process based on user inputs thus far.

U.S. Patent 5,802,255 teaches a user interface design that enables building rules in the area of intelligent interaction with rules. However, the rules taught are specifically condition-action rules and are thus highly structured.

U.S. Patent 5,596,502 is a narrow manufacturing application. U.S. Patent 5,444,632 describes factory lot processing and scheduling methods.

OBJECTS AND ADVANTAGES OF THE INVENTION

Accordingly, it is a primary object of the present invention to overcome the limitations of the prior art rule building interface methods. More specifically, it is an object of the invention to provide a fast, efficient and flexible computer-implemented method to specify constraints and other business rules to be considered in a scheduling system.

SUMMARY OF THE INVENTION

These objects and advantages are attained by a novel computer-implemented method for improving a person's schedule. The method is distinguished from the prior art by numerous advantageous features. For example, the method is more versatile than the prior art, because of inclusion of two input devices, and because of the fact that the resulting rules are not in fact forced to satisfy a particular grammar. The ability to express self-referential and abstract qualities are also improvements over existing solutions. Further, many existing methods require a database as part of the essential mechanism, whereas the interface of the present invention does not in principle require a database.

The present invention is an interface system for interactively building rules and constraints. The interface system comprises of an output display configured to show a user a partial rule with a current rule fragment, such rule fragment comprising a blank space; the output display is further configured to show a user a list of potential selections for filling the blank space in the current rule fragment; the output display also shows a user a completed rule wherein the completed rule should be easily parsed by a human user; and at least one input device configured to allow a user to choose from a list a desired selection for filling the blank space and for allowing a user to directly enter values to create a selection for filling the blank space, thereby creating a completed rule. The input device can also assign a tolerance to the completed rule, whereby the completed rule is made self-referential. A completed self-referential rule refers to a goal that is unspecified in an absolute sense or to a schedule that does not yet exist.

The interface system further comprises a processing element configured to apply branching rules to previous selections of a user for filling the blank space, thereby interactively and dynamically creating future blank spaces and future lists of the potential selections. Alternatively, the interface system can include a processing element configured to access a dynamic database, thereby to populate the lists of potential selections depending on the current value in real time of the dynamic database.

The interface system further comprising a processing element configured for converting a completed rule into an internal representation suitable for inputting into a resource scheduling system. The interface system also assigns the completed rule to at least one person in a resource schedule.

The interface system of the present invention can be used for resource scheduling of call center agents in a call center as well as other resource scheduling applications.

DESCRIPTION OF THE FIGURES

Fig. **1** is a block diagram indicating an interface system for building rules and constraints for a resource scheduling system, in accordance with the present invention; and

Fig. **2** is a flow diagram illustrating the operation of an embodiment of the interface system according to Fig. **1**.

DETAILED DESCRIPTION

Although the following detailed description contains many specifics for the purposes of illustration, anyone of ordinary skill in the art will appreciate that many variations and alterations to the following details are within the scope of the invention. Accordingly, the following preferred embodiment of the invention is set forth without any loss of generality to, and without imposing limitations upon, the claimed invention.

One embodiment of the present invention involves unscheduling one or more personnel from their shifts in the schedule, so that their shifts are empty. These personnel, or a portion of these personnel are then rescheduled for different possible shifts so that various possible schedules are formed. Each of these possible schedules are evaluated with a score function and based on the scores, an improved schedule is selected from the possible schedules. The same process can also be applied to find an optimal schedule for breaks.

At the highest level, our method differs in that the system we propose is creating rules and not sentences in a grammar. One may argue that as long as the set of possible rules is finite, then this defines a grammar. Our set of rules differs depending on the state of the database and, therefore, it is not a fixed language. Furthermore, we improve on the method and specify by creating a self-referential component in the rules (fairness).

And, we further improve by having not only some blanks filled by the user through selection from a list, but by enabling other blanks to be filled by the user through direction specification, without the need to choose from a list. These improvements make our method more expressive because it can more flexibly create an array of rules without being tied to one grammar.

Fig. **1** is a block diagram indicating an interface system **100** for building rules and constraints for a resource scheduling system, in accordance with the present invention. In an embodiment, interface system **100** incorporates an output display **102** for showing a user a current rule fragment having a blank space to be filled in by a user, thereby completing the rule. Output display **102** is interconnected through a wired or wireless data link **110** with an interface system processor **104**. Interface system **100** contains input devices **106** and **108** configured for allowing the user to fill in the blank space. A first input device **106** allows the user to fill in the blank space by selecting from a list of potential selections displayed on output display **102**. A second input device **108** allows the user to fill in the blank space by direct entry of open-ended values. An optional input device **112** allows the user to specify a self-referential constraint or tolerance (e.g., a rule is X and this rule must be complied within a tolerance of Y). Such a self-referential rule can refer to a goal that is unspecified in an absolute sense or to a schedule that does not yet exist. The function of optional input device **112** can alternatively be incorporated into either or both of the other input devices **106** and **108**. The completed rule with the blank space filled is displayed at output display **102**.

Interface system **100** is interactive, applying branching rules to previous selections of a user to create future blank spaces and future lists of potential selections. In some embodiments, interface system **100** includes a processing element that accesses a dynamic database **114**, such that lists of potential selections are populated in accordance with the current value in real time of the dynamic database. In some embodiments, interface system **100** includes a processing element **116** for converting a completed rule into an internal representation suitable for inputting into a resource schedule system **118**, e.g., a call center scheduling system. Processing element **116** can alternatively be included as part of interface system processor **104**. A processing element **120** for assigning a completed rule to a person **124**, e.g., a call center person, in a resource schedule **118**, is optionally provided, either separately or contained within interface system processor **104**.

In some embodiments, interface system **100** and resource scheduling system **118** are integrated into a common scheduling system **130**, for example a call center scheduling system.

In an interface system according to the present invention, rules that result from the interface are easily parsed by a human user, but are not forced to follow any particular grammar. In some cases, the human user chooses an appropriate term from a list to fill a blank space, whereas in other cases, the human user creates a term to fill the blank space, without being bound to a list. In both of the above types of operation, which blanks exist and what terms are contained in future choice lists are guided dynamically by the user's former choices. The process includes an element that uses fixed branching rules, and an element that goes to a dynamic database and populates the lists depending on the current value of the database in real time. Part of the interface system is designed for the user to optionally specify a tolerance with which he wishes the scheduler to abide by the rule. In other words, the rules can be self-referential (e.g., "Joe works four times per month, with a tolerance of one"; "All employees work equal number of Saturday night shifts, with a tolerance of three.") The constraints can also be abstract and referential in the sense that they can refer to a schedule that does not yet exist (e.g., this rule is that everyone according to the final schedule should work on average the same number of Mondays.) This rule refers to the schedule (solution) that does not yet exist, without specifying a goal in an absolute sense. Further, the interface is designed such that these rules can first be created, then assigned or attached to one or more individual personnel (or objects of the rule).

Fig. **2** is a flow diagram **200** illustrating one example of the operation of an embodiment of interface system **100**. In the example of Fig. **2**, at step **202** the user selects either "Employee must work…," transferring operation to step **204**; or "Employee must have off…," which transfers operation to step **214**. Steps **204** and **214** invite the user to fill in a value in an appropriate blank space. Steps **206** and **216** provide choices and blank fill-in spaces to select calendar and/or time units. At step **206**, the shifts (shift patterns) selection provides the user with a dynamically populated list of selections based on the current state of the database. Operation shifts from steps **206** and **216** respectively to steps **208** and **218**, providing the user a choice of portions of the day before transferring operation to respective steps **210** and **220**. Steps **210** and **220** provide a choice of repeat time intervals. Alternatively, step **208** and/or **218** are bypassed, providing operational flow directly from step **206** to step **210**, or from step **216** to **220**. The user can then direct operational flow from step **210** and/or step **220** to step **212**, where tolerances are applied (for fairness rules only). Thereafter the user selects the start of the schedule at step **222**. Alternatively, the user can bypass step **212** and execute step **222** directly after step **210** or step **220**. The above operations illustrate the basic flow of the present invention. Other operations may be added, subtracted, or substituted from the basic flow depending on the specific problem situation.

It will be clear to one skilled in the art that the above embodiment may be altered or supplemented in many ways without departing from the scope of the invention. Additional variations will be obvious to those skilled in the art. Accordingly, the scope of the invention should be determined by the following claims and their legal equivalents.